基于变异交叉方程与进化选择机制的回溯优化改进算法 *

赵琳敬 [1a, 1b]，葛宝臻 [1a, 1b]，陈　雷 [1a, 1b, 2†]

(1. 天津大学 a. 精密仪器与光电子工程学院，天津 300072; b. 光电信息技术教育部重点实验室，天津 300072; 2. 天津商业大学 信息工程学院，天津 300134)

摘　要：针对回溯搜索优化算法存在的收敛速度慢，容易陷入局部最优等问题，提出了一种改进算法。首先利用 t 分布产生变异尺度系数，加快了算法收敛速度；接着完善交叉方程结构，引入最优个体控制种群搜索方向，有效提高了算法开发能力；最后提出进化选择机制，引入差分进化算法变异因子，一定概率下以较差解替换较优解，避免算法陷入局部最优。在数值实验中，选取了 15 个测试函数进行仿真测试，并与 5 种表现良好的算法进行了比较，结果表明，该算法在收敛速度及搜索精度方面有明显优势。

关键词：回溯搜索优化算法；变异方程；交叉方程；差分进化算法

中图分类号：TP301.6　　doi: 10.3969/j.issn.1001-3695.2017.12.0848

Backtracking search optimization algorithm based on mutation and crossing equations and evolutionary selection mechanism

Zhao Linjing[1a, 1b], Ge Baozhen[1a, 1b], Chen Lei[1a, 1b, 2†]

(1. *a. School of Precision Instruments & Opto-Electronics Engineering*, *b. Key Laboratory of Opto-Electronics Information & Technical Science of Ministry of Education*, *Tianjin University*, *Tianjin* 300072, *China*; 2. *School of Information Engineering*, *Tianjin University of Commerce*, *Tianjin* 300134, *China*)

Abstract: According to the slow convergence and easiness to trap in local optimum of backtracking search optimization algorithm, this paper presented an improved algorithm. The method proposed a mutation scale factor based on t distribution firstly to speed up the convergence rate; then the algorithm improved the structure of crossover equation and introduced the optimal individual to control the direction of population search, which effectively improved the development capability. Finally, the algorithm proposed the evolutionary selection mechanism, introduced the mutation factor of differential evolution algorithm and replaced the optimal solution with worse solution under a certain probability, which can avoid algorithm to fall into the local optimum. The numerical experiments selected 15 test functions for simulation and compared with 5 well-behaved algorithms. The results show that the proposed algorithm has obvious advantages in terms of convergence rate and search accuracy.

Key words: backtracking search optimization algorithm(BSA); mutation equation; crossing equation; differential evolution

0　引言

近些年，进化算法以其结构简单、参数较少、易于实现等特点受到广泛应用，各种进化算法层现叠出，如经典的遗传算法（genetic algorithm, GA）[1]、粒子群优化算法（particle swarm optimization, PSO）[2]、差分进化算法（deferential evolution, DE）[3]、人工蜂群算法（artificial bee colony algorithm, ABC）[4]等。然而随着高维多模态复杂优化问题出现的愈加频繁，使全局优化算法面临着更大的挑战。对此，学者们在现有进化算法的基础上，提出了大量改进算法以及新的优化算法。例如，Zhu 等人[5]提出的蜂群改进算法 GABC，李文斌等人[6]提出的差分进化改进算法 CNS-DE，姜凤利等人[7]提出的粒子群优化改进算法 IPSO，以及 Meng 等人[8]提出的鸡群算法（chicken swarm optimization, CSO），Zheng 等人[9]提出的水波算法（water wave optimization, WWO），高江锦等人[10]提出的免疫否定进化选择算法（immune evolution negative selection algorithm,IENSA）等新型进化算法。

回溯搜索优化算法（backtracking search algorithm, BSA）是 Civicioglu[11]于 2013 年提出的一种新型进化算法。算法框架与差分进化算法类似，但具体操作有本质区别[12]。BSA 采用历史

收稿日期：2017-12-08；**修回日期：**2018-02-26　　**基金项目：**国家自然科学基金重点资助项目（61535008）

作者简介：赵琳敬（1994-），女，河北石家庄人，硕士研究生，主要研究方向为进化算法、计算机视觉；葛宝臻（1964-），男，内蒙古卓资人，教授，博士，主要研究方向为光电检测与信息处理、激光粒子测量、激光三维彩色数字化技术；陈雷（1980-），男（通信作者），河北唐山人，副教授，博士，主要研究方向为盲信号处理和智能计算（chenlei@tjcu.edu.cn）.

种群控制搜索方向，加之其高效新颖的混合交叉策略，使其各项性能都取得了不错的成绩。

但整体来看，BSA 探索能力较强而开发能力有待提高，导致收敛速度较慢。针对这一缺点，本文提出了基于 t 分布的变异系数 F，以及加入最优个体引导的新交叉方程，在保证探索能力的基础上，大大提高了算法的收敛速度。同时，提出进化选择机制，引入差分进化算法变异算子 DE/rand/1，增加全局探索能力，避免算法早熟收敛。最后，选取 15 个测试函数，对包括本文改进算法在内的六种算法进行数值实验，结果表明，本文改进算法显著提高了 BSA 的收敛速度和精度。

1 回溯搜索优化算法介绍

BSA 是一种基于群体智能的进化算法，算法结构与其他进化算法类似，主要包括五个步骤，分别为种群初始化、选择Ⅰ、种群变异、种群交叉、选择Ⅱ。

1）种群初始化

BSA 初始化种群的方法为随机产生，方程如式(1)(2)所示。

$$\boldsymbol{Pop}_{i,j} \sim U(\boldsymbol{Low}_j, \boldsymbol{Up}_j) \tag{1}$$

$$\boldsymbol{Oldpop}_{i,j} \sim U(\boldsymbol{Low}_j, \boldsymbol{Up}_j) \tag{2}$$

其中：$\boldsymbol{Pop}$ 为初始种群；$\boldsymbol{Oldpop}$ 为历史种群；$i \in \{1,2\cdots N\}$，$j \in \{1,2\cdots D\}$；N 是种群个数；D 是种群维数；Low 和 Up 分别为搜索空间的下界和上界；U 为随机均匀分布函数。

2）选择Ⅰ

BSA 通过历史种群控制搜索方向，选择Ⅰ的目的是在每次迭代开始时选择新的历史种群。选择策略如式(3)所示。

$$if\ a < b\ then\ \boldsymbol{Oldpop} = \boldsymbol{Pop} \tag{3}$$

其中：a、b 是（0,1）上服从均匀分布的随机数。在确定 $\boldsymbol{Oldpop}$ 之后，将 $\boldsymbol{Oldpop}$ 中种群的位置进行重新排列，公式如下：

$$\boldsymbol{Oldpop} = \mathrm{randperm}(\boldsymbol{Oldpop}) \tag{4}$$

其中：randperm 是重新排序函数。

3）变异

BSA 的变异方程如(5)所示。

$$\boldsymbol{M} = \boldsymbol{Pop} + F \cdot (\boldsymbol{Oldpop} - \boldsymbol{Pop}) \tag{5}$$

其中：F 为变异尺度系数用以控制搜索方向矩阵 $(\boldsymbol{Oldpop} - \boldsymbol{Pop})$ 的幅度，$F = 3 * \mathrm{randn}$，randn 是一服从标准正态分布的随机数。

4）交叉

BSA 提出了一种新型的交叉策略，通过设置混合比例参数来控制种群粒子交叉个数。具体公式如式(6)所示。

$$T_{i,j} = \begin{cases} \boldsymbol{M}_{i,j}, & \boldsymbol{Map}_{i,j} = 1 \\ \boldsymbol{Pop}_{i,j}, & \boldsymbol{Map}_{i,j} = 0 \end{cases} \tag{6}$$

其中：$\boldsymbol{Map}_{i,j}$ 为 $N \times D$ 的二元整数矩阵；初始赋值为 1。具体公式如式(7)如示。

$$\begin{cases} \boldsymbol{Map}_{i,(u\lceil 1:\mathrm{mixrate}\cdot \mathrm{rand}\cdot D\rceil)} = 0, & a < b \\ \boldsymbol{Map}_{i,randi(D)} = 0, & a \ge b \end{cases} \tag{7}$$

其中：randi(D) 为从[0，D]中随机取一个整数，rand；a、b 为[0,1]的随机数；mixrate 为交叉概率；$\mathrm{mixrate} \cdot \mathrm{rand} \cdot D$ 为交叉长度；$u = \mathrm{randperm}(D)$，为 $[1,2\cdots D]$ 重新排序后的整数向量。BSA 通过调用这两种等概率的 $\boldsymbol{Map}$ 产生方式，有效控制新种群 $\boldsymbol{T}$ 中变异元素的个数。

在新种群 T 确定后，对种群中的元素进行边界检测，若超出搜索范围，则按式(1)产生新的个体。

5）选择Ⅱ

通过贪婪法则在新种群与初始种群中选择适应度值较好的个体，并记录当前全局最优解和对应的解向量，同时更新初始种群。更新方程如式(8)所示。

$$\boldsymbol{Pop}_i = \begin{cases} \boldsymbol{T}_i & fitness(\boldsymbol{T}_i) < fitness(\boldsymbol{Pop}_i) \\ \boldsymbol{Pop}_i & otherwise \end{cases} \tag{8}$$

重复上述过程，直到满足终止条件，最后输出最优解。

2 BSA 改进算法

BSA 算法全局探索能力较强而局部开发能力稍显弱势，究其原因是该算法仅在历史种群的引导下进行变异，缺少最优个体的引导。针对这一特点，本文对变异及交叉方程进行了改进；另一方面，本文提出了算法进化率 P，引入差分进化变异因子，使算法后期种群多样化，避免算法陷入局部最优。本文将所提出的改进算法简记为 MEBSA。

2.1 变异尺度系数改进

原始 BSA 算法的变异尺度系数 $F = 3 * randn$，分布范围过广，容易产生较大或较小的极值，不利于种群的快速收敛。为此，本文提出了一种基于 t 分布的变异尺度系数。其表达式为

$$F = 2 * \sqrt{\mathrm{trnd}(df)} \tag{9}$$

其中：trnd(df)是一服从 t 分布的随机数。

t 分布只有一个参数，即自由度 df，其密度函数较于标准正态分布的密度函数表现为厚尾，且其形状随着 df 的增大而无限接近于正态分布。因此，相比于正态分布，通过自由度 df，可以很方便地控制 t 分布的形状，且因其厚尾的特点，在整体分布趋势较为集中的基础上，也可以一定概率取到处于边界的极值，从而避免算法过分收敛，陷入局部最优。本文取 df=10。基于 t 分布的变异尺度系数与原始系数绝对值比较，在不同区间的简要分布情况如表 1 所示。

表 1 各区间简要分布概率

区间	(0,0.5)	(0.5,1)	(1,1.5)	(1.5,2)	(2,4)	>4
$2*\sqrt{\mathrm{trnd}(df)}$	0.0487	0.1412	0.2229	0.2447	0.3404	0.0021
$3*\mathrm{randn}$	0.132	0.128	0.121	0.112	0.323	0.1824

由表 1 可以看出，与原始系数相比，新的变异系数产生的扰动较为集中，且不会产生过大的数值，加快了种群的收敛速度，一定程度上提高了算法的局部开发能力。

2.2 交叉方程改进

通过对 BSA 算法的分析可知，该算法因其独特的变异交叉操作，全局探索能力强，但因其仅在历史种群的引导下控制搜

索方向，导致该算法的收敛速度较慢，局部开发能力还不够强。针对这一缺点，本文算法引入最优个体的引导，提出了一个新的交叉方程，公式如式(10)(11)所示。

$$T_{i,j}=\begin{cases}M_{i,j}+C(Gpop_{i,j}-Pop_{i,j}), & Map_{i,j}=1\\ Pop_{i,j}+C(Gpop_{i,j}-Pop_{i,j}), & Map_{i,j}=0\end{cases} \tag{10}$$

$$Gpop=\text{rempat}((\text{normrnd}(1,1).*Gminimizer),N) \tag{11}$$

其中：C 为一服从标准正态分布的随机数；$Gpop$ 是一个 $N\times D$ 矩阵，矩阵元素由当前种群最优个体的正态分布组成。这样一来，BSA 算法的交叉变异方程趋于完整，不仅有历史种群的引导，还有当前最优个体控制搜索方向。值得一提的是，本文没有直接采用当前全局最优个体引导，而是加上了正态分布的扰动，使算法在大大提高了收敛速度之外，也保证了全局探索能力。

2.3 进化选择机制

原始 BSA 算法在初期迭代中具有较好的探索能力，然而随着循环次数的增多，仍然面临着早熟收敛的问题，对于较复杂的多峰多极值函数，存在可能无法搜索到最优值的缺点。因此，本文提出了进化选择机制，引入算法进化率 P，设定每隔 per 代，计算一次 P。当 P 小于所设阈值 ε 时，随机选取 Q 个当前个体，引入差分进化算法变异算子 DE/rand/1，产生 Q 个新的个体，并替换掉当前种群及历史种群中的相应个体，以一定的概率将较差的解代替较优解，使算法中后期的种群更加多样化，避免早熟收敛。新个体产生公式如式(12)所示。

$$Newpop_i=pop_{v(i)}+W(Pop_{r1}-Pop_{r2}) \tag{12}$$

其中：$i\in\{1,2,\cdots Q\}$，$v(i)=cr(i)$，$cr=\text{randperm}(N)$；W 取[-1,1]的随机数；$r1\neq r2\in\{1,2,\cdots N\}$ 且互不相同。本文算法设定 per=100，ε =0.5，$Q=\text{ceil}(0.3*N)$。其中：ceil 为向上取整函数。下面给出改进 BSA 算法的流程：

```
种群初始化
while 不满足算法停止条件 do
进行选择 I，选取 Oldpop
if rand<rand do
  Map_{i,(u⌈1:mixrate·rand·D⌉)} = 0
else
  Map_{i,randi(D)} = 0
end if
按式(10)进行变异操作
按式(6)进行交叉操作
if P<0.5
  for i=1: ceil(0.3*N)
    按式(12)产生新的个体，并替换
  end for
end if
进行选择 II，更新 Pop
end while
输出最优解
```

3 数值实验及结果分析

为了测试本文改进算法的效果，选取了 15 个测试函数进行数值实验，同时与标准 BSA、BSA 改进算法 COBSA[13]、DS[14]、MABC 算法[15]以及 CSO 算法进行了对比实验。表 2 给出了 15 个测试函数的名称、维数、搜索空间和最优

值[16,17]。其中，F1~F7 是简单的单峰测试函数；F8~F9 是多峰少极值函数；F10~F15 是多峰多极值函数。这些函数的局部极值个数随着函数维数的增加呈指数级增长，这类函数对大多数算法都是很难求解的。

表 2　15 个函数的简单描述

函数	名称	维数	搜索空间	最优值
F1	Sphere	60	[-100,100]	0
F2	Schwefel 2.22	60	[-10,10]	0
F3	Sumsquare	60	[-10,10]	0
F4	Exponential	60	[-1.28,1.28]	0
F5	Schwefel 1.2	10	[-30,30]	0
F6	Elliptic	30	[-100,100]	0
F7	R-H-Ellipsoid	80	[-100,100]	0
F8	Zakharov	60	[-5,10]	0
F9	Salomon	60	[-100,100]	0
F10	Alpine	60	[-10,10]	0
F11	Rastrigin	100	[-100,100]	0
F12	Griewanlk	60	[-600,600]	0
F13	Schaffer	10	[-100,100]	0
F14	Weierstrass	60	[-0.5,0.5]	0
F15	NCRastrigin	50	[-5.12,5.12]	0

3.1 停止条件制定及参数设定

（a）所求结果与测试函数的理论最优解绝对值之差小于 Matlab 系统最小正浮点数 4.9407e-324，即 eps（0）时，停止。

（b）达到最大进化代数 epoch 时，停止。

达到最大进化代数之前算法终止，则本次运行结果为成功。实验种群数目取 60，epoch 取 50 000。实验均在相同配置的计算机上，使用 MATLAB 2013b 进行编程测试。

3.2 测试结果分析

针对同一测试函数，每种算法独立运行 30 次，比较六种算法对 15 个标准测试函数结果的统计平均值（mean）、方差（std）、平均迭代次数（miter）及成功率（srate）。测试结果如表 3 所示。从表 3 中可以看出，在 30 次的独立运行中，无论是在单峰测试函数、多峰少极值函数还是复杂的多峰多极值函数上，MEBSA 都能达到预设精度，且迭代次数远远小于其他五种算法。充分证明了 MEBSA 不仅具有较快的收敛速度和强鲁棒性，还有较强的全局搜索性能，能够快速跳出局部最优，避免早熟收敛。

表 3　六种算法对 15 个测试函数的测试结果

问题	统计量	BSA	DS	CSO	COBSA	MABC	MEBSA
F1	mean	3.04e-158	9.12e-257	0	0	5.01e-16	0
	std	9.02e-158	0	0	0	5.83e-17	0
	miter	50000	50000	23003	10538	50000	2356
	srate	0	0	0.93	1	0	1
F2	mean	2.22e-95	3.99e-143	0	0	8.51e-16	0
	std	3.70e-95	1.22e-142	0	0	1.10e-16	0
	miter	50000	50000	13851	15496	50000	4045
	srate	0	0	1	1	0	1
F3	mean	1.56e-158	1.69e-257	0	0	5.19e-16	0
	std	5.08e-158	0	0	0	6.46e-17	0
	miter	50000	50000	20375	10452	50000	2158
	srate	0	0	1	1	0	1
F4	mean	1.44e-96	6.27e-145	0	0	648.6356	0
	std	2.39e-96	2.89e-144	0	0	44.0127	0
	miter	50000	50000	13845	15347	50000	4082
	srate	0	0	1	1	0	1
F5	mean	0.0020	0.0086	1.46e+03	0	2.4568e+03	0
	std	0.0014	0.0094	505.9816	0	299.1724	0
	miter	50000	50000	50000	27097	50000	3924
	srate	0	0	0	1	0	1
F6	mean	0	0	0	0	8.11e-16	0
	std	0	0	0	0	8.60e-17	0
	miter	48359	38504	6752	10521	50000	2510
	srate	1	1	1	1	0	1
F7	mean	1.85e-158	5.22e-256	0	0	5.23e-16	0
	std	9.33e-158	0	0	0	6.84e-17	0
	miter	50000	50000	16510	11087	50000	2204
	srate	0	0	0.8	1	0	1
F8	mean	1.07e-05	2.67e-06	0.85	2.48e-211	8.87e-16	0
	std	1.12e-05	1.44e-06	2.69	0	1.32e-16	0
	miter	50000	50000	50000	50000	50000	3825
	srate	0	0	0	0	0	1
F9	mean	0.6632	0.3699	0.2409	0.0999	2.4199	0
	std	0.1129	0.0535	0.0555	2.31e-12	0.1375	0
	miter	50000	50000	50000	50000	50000	2277
	srate	0	0	0	0	0	1
F10	mean	2.45e-13	9.84e-52	0.0095	0	1.77e-15	0
	std	1.32e-12	5.15e-51	0.0153	0	1.47e-15	0
	miter	50000	50000	30202	15461	50000	4121
	srate	0	0	0.57	1	0	1
F11	mean	0.0995	0.1658	24.5163	0	0	0
	std	0.3036	0.3771	94.3434	0	0	0
	miter	33252	13523	10757	661	9703	158
	srate	0.5	0.83	0.83	1	1	1
F12	mean	2.47e-04	0	1.11e-17	0	1.07e-16	0
	std	0.0014	0	3.39e-17	0	1.07e-16	0
	miter	7379	3864	9803	698	40922	165
	srate	0.97	1	0.90	1	0.33	1
F13	mean	0	0	0	0	0.9550	0
	std	0	0	0	0	0.0794	0
	miter	29357	21685	3039	7938	50000	2930
	srate	1	1	1	1	0	1
F14	mean	0	0	2.37e-15	0	0	0
	std	0	0	9.20e-15	0	0	0
	miter	8827	5022	5367	765	3454	170
	srate	1	1	0.93	1	1	1
F15	mean	5.30e-14	0	1.89e-14	0	3.79e-15	0
	std	6.49e-14	0	5.24e-14	0	2.07e-14	0
	miter	31886	6438	9656	652	15839	162
	srate	0.57	1	0.87	1	0.97	1

为了进一步直观说明 MEBSA 算法的先进性，图 1~4 给出了典型的四种测试函数的收敛曲线。其中横坐标为迭代次数，纵坐标为函数值。(注:为了方便进化曲线的显示和观察,本文对函数值取以 10 为底的对数)。

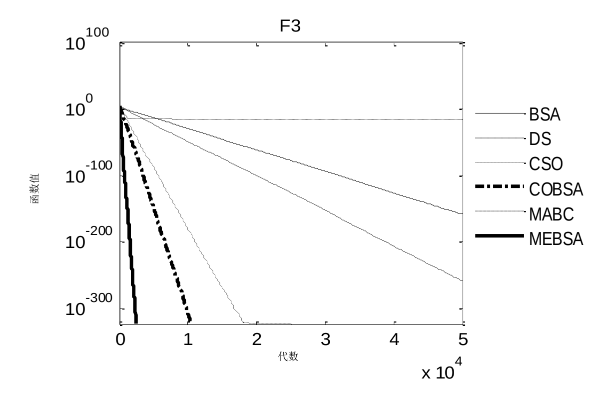


图 1　F3 函数的收敛曲线

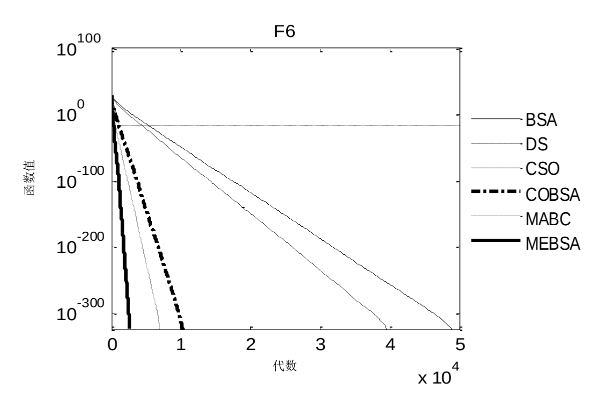


图 2　F6 函数的收敛曲线

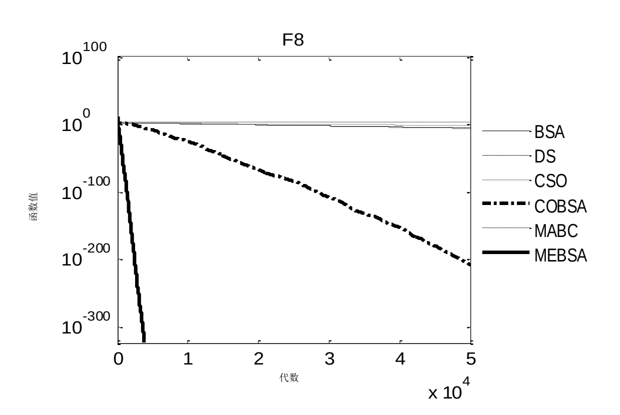


图 3　F8 函数的收敛曲线

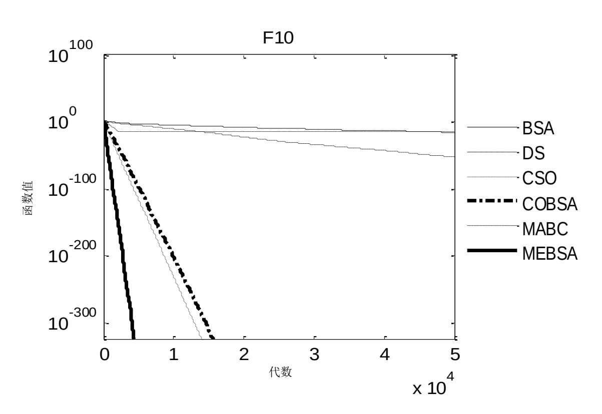


图 4　F10 函数的收敛曲线

结合图表可以看出，针对 F3、F6、F10 等其他五种算法也能部分成功运行的函数，MEBSA 能以最少的迭代次数以及最快的收敛速度达到预设精度；针对 F8、F9 等其他五种算法均不能成功的函数，MEBSA 依然能以较快速度成功收敛。进一步展示了 MEBSA 算法的优越性。

4 结束语

为了解决标准 BSA 算法收敛速度相对较慢及搜索精度不高的问题，本文提出了新的变异尺度系数 F，并加入最优个体的引导，构造了一个新的交叉方程；同时，为了防止算法陷入局部最优，提出了进化选择机制，引入差分进化变异算子。最后，选取 15 个测试函数，并与五种算法进行了对比。结果表明，本文改进算法 MEBSA 有效提高了算法的各项搜索性能。

参考文献：

[1] Holland J H. Adaptation in natural and artificial systems [M]. [S. l.] : Ann Arbor. University of Michigan Press, 1975.

[2] Eberhart R, Kennedy J. A new optimizer using particle swarm theory [C]// Proc of Micro Machine and Human Science. Washington DC: IEEE Computer Society, 1995: 39-43.

[3] Storn R, Price K. Differential evolution: a simple and efficient heuristic for global optimization over continuous spaces [J]. Journal of Global Optimization, 1997, 11 (4): 341-359.

[4] Karboga D, Basturk B. A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm [J]. Journal of Global Optimization, 2007, 39 (3): 459-471.

[5] Zhu Guopu, Kwong S. Gbest-guided artificial bee colony algorithm for numerical function optimization [J]. Applied Mathematics & Computation, 2010, 217 (7): 3166-3173.

[6] 李文斌，陈嶷瑛，贺毅朝，等. 邻域结构为复杂网络的差分演化算法 [J]. 计算机应用研究, 2016, 33 (2): 370-374.

[7] 姜凤利，张宇，王永刚，等. 一种基于引导策略的自适应粒子群优化算法 [J]. 计算机应用研究, 2017, 34 (12): 3599-3602.

[8] Meng Xianbing, Liu Yu, Gao Xiaozhi, *et al.* A new bio-inspired algorithm: chicken swarm optimization [C]// Proc of International Conference on Swarm Intelligence. Hefei: Springer International Publishing, 2014: 86-94.

[9] Zheng Yujun. Water wave optimization: a new nature-inspired metaheuristic [J]. Computers and Operations Research, 2015, 55: 1-11.

[10] 高江锦，杨韬. 免疫进化否定选择算法 [J]. 计算机应用研究, 2017, 34 (5): 1293-1297.

[11] Civicioglu P. Backtracking search optimization algorithm for numerical optimization problems [J]. Applied Mathematics and Computation, 2013, 219 (15): 8121-8144.

[12] 田文凯，刘三阳，王晓娟. 基于差分进化的回溯搜索优化算法研究与改进 [J]. 计算机应用研究, 2015, 32 (6): 1653-1656.

[13] Zhao Lei, Jia Zhicheng, Chen Lei, *et al.* Improved backtracking search algorithm based on population control factor and optimal learning strategy [J]. Mathematical Problems in Engineering, 2017, 5 (1): 26-36.

[14] Civicioglu P. Transforming geocentric cartesian coordinates to geodetic coordinates by using differential search algorithm [J]. Computers and Geosciences, 2012, 46 (3): 229-247.

[15] Gao Weifeng, Liu Sanyang. A modified artificial bee colony algorithm [J]. Computers and Operations Research, 2012, 39 (3): 687-697.

[16] Liang Jing, Qin A K, Suganthan P N, *et al.* Comprehensive learning particle swarm optimizer for global optimization of multimodal functions [J]. IEEE Trans on Evolutionary Computation, 2006, 10 (3): 281-295.

[17] Karaboga D, Akay B. A comparative study of artificial bee colony algorithm [J]. Applied Mathematics and Computation, 2009, 214 (1): 108-132.